Available online at www.sciencedirect.com
ScienceDirect
Computers and Mathematics with Applications 54 (2007) 161–182
An International Journal
computers & mathematics
with applications
www.elsevier.com/locate/camwa

Simulation of IPA gradients in hybrid network systems

Benjamin Melamed[a,*], Shuo Pan[b], Yorai Wardi[c]

[a] *Rutgers University, Rutgers Business School - Newark and New Brunswick, Department of MSIS, United States*
[b] *Rutgers University, Rutgers Center for Operations Research, United States*
[c] *Georgia Institute of Technology, School of Electrical and Computer Engineering, United States*

Received 9 October 2006; received in revised form 1 December 2006; accepted 12 December 2006

Abstract

Infinitesimal perturbation analysis (IPA) provides formulas for random gradients (derivatives) of performance measures with respect to parameters of interest, computed from sample paths of stochastic systems. In practice, IPA derivatives may be computed either from simulation runs or from empirical field data (when the formulas are nonparametric). Nonparametric IPA derivatives in fluid-flow queues have been recently derived for the loss volume and time average of buffer occupancy, with respect to buffer size, and arrival-rate or service-rate parameters. Additionally, these IPA derivatives have been shown to be unbiased in the sense that their expectation and differentiation operators commute, while their traditional discrete counterparts have long been known to be generally biased. Recent work has further shown how to map the computation of IPA derivatives from a fluid-flow queue to a compatible discrete counterpart without an appreciable loss of accuracy in performance measures. Thus, this work holds the promise of potential applications of IPA derivatives to gradient-based optimization of objective functions involving performance metrics parameterized by settable parameters in a queueing network context.

This paper is an empirical study of IPA derivatives of individual queues within queueing systems which model telecommunications networks and some of their protocols. As a testbed, we used HNS (Hybrid Network Simulator) — a hybrid Java simulator of queueing networks with traffic streams subject to several telecommunications protocols. More specifically, the hybrid feature of HNS admits models with mixtures of discrete (packet) flows and continuous (fluid) flows, and collects detailed statistics and IPA derivatives for all flow types. The paper outlines the mapping of IPA derivatives from the fluid domain to the packet domain as implemented in HNS, and studies the accuracy of IPA derivatives in compatible fluid and packet queueing models, as well as the stabilization of their values in time. Our experimental results lend empirical support to the contention that IPA derivatives can be accurately computed from discrete versions by adopting a fluid-flow view. Furthermore, the long-run values of various IPA derivatives are empirically shown to stabilize quite fast. Finally, the results provide the basis and motivation for IPA applications to the optimization of telecommunications network design and to potential new open-loop protocols that take advantage of IPA information.

General Terms: Algorithms; Design; Performance

Keywords: Fluid-flow models; Fluid-flow simulation; Hybrid simulation; Infinitesimal perturbation analysis; IPA; IPA derivatives; IPA gradients; Packet models

* Corresponding address: Department of MSIS, Rutgers Business School - Newark and New Brunswick, Rutgers University, 256 Levin Building, 94 Rockafeller Road, Piscataway, NJ 08854-8054, United States.
E-mail address: melamed@rbs.rutgers.edu (B. Melamed).

0898-1221/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2006.12.032

1. Introduction

Dynamic Monte Carlo simulation methods are widely used in analyzing complex queueing networks when current analytical methods cannot provide closed-form or numerical solutions [1,2]. Traditionally, the vast majority of queueing systems have been modeled in terms of discrete workload units that arrive at and depart from queue buffers in an "abrupt" manner. In contrast, the fluid-flow variety of queueing systems treats workload as fluid that flows "gradually" into and out of queue buffers at some (possibly random) rates. We will refer to the former queueing systems as ***discrete*** or ***traditional*** and to the latter as ***continuous*** or ***fluid-flow*** (sometimes abbreviated to just ***fluid***). The terms ***gradient*** and ***derivative*** will be used interchangeably throughout the paper.

In addition to fluid-flow analytical models for queueing networks (e.g., [3,4]), fluid models have been proposed for telecommunications networks under specific protocols [5]. For example, fluid simulation for ATM networks [6] is discussed in [7]. This model describes a fluid event-driven ATM network simulator that uses Markov-modulated fluid models for the sources, as well as fluid leaky buckets and fluid bandwidth schedulers. An overview of fluid-flow network modeling can be found in [8], including fluid versions of four common workload-processing schedulers: work-conserving generalized processor sharing (GPS) schedulers, non work-conserving idling schedulers, FIFO schedulers, and priority schedulers. A fluid-flow model for the TCP protocol is proposed in [9]. This model captures the TCP congestion control mechanism both in slow start and congestion avoidance modes, as well as acknowledgements, lost traffic, timeouts, and retransmissions. Finally, [10] compares the simulation complexity of packet-based and fluid-based simulations.

Standard queueing processes of interest include buffer occupancy, lost workload and job sojourn times, and associated performance measures are often formulated as time averages and means. ***Infinitesimal perturbation analysis*** (***IPA***) augments such statistics with sensitivity information. More specifically, IPA is a sample-path technique for computing gradients of performance metrics with respect to design/control parameters of interest, such as buffer size and parameters of service and arrival processes [11,12]. Formally, let $L(\theta)$ be a real-valued random variable that depends on a real parameter $\theta \in \Theta$, where Θ is a real set. Let further $l(\theta) = E[L(\theta)]$ be the parameter-dependent expectation function. The ***IPA derivative*** (***gradient***) of $L(\theta)$ is the random variable $L'(\theta) = \frac{d}{d\theta} L(\theta)$, provided that it exists. If $E[L'(\theta)] = l'(\theta)$, then the IPA derivative is said to be ***unbiased***.

In principle, IPA derivatives can provide a basis for research on design optimization and control applications for simulated systems, since simulation-based mean-derivative estimates can be used to optimize objective functions formulated in terms of performance metrics of interest. Such objective functions are often expressed as cost functions associated to performance metrics, such as link loss rate, and time average of link buffer occupancy (or, equivalently, of mean waiting time, by Little's formula [13]). IPA derivatives can then be used in simulation-based gradient-driven techniques to optimize system performance. Moreover, if the IPA derivatives are ***nonparametric*** (that is, they can be derived without making distributional assumptions on the underlying random processes), then the aforementioned approaches can be applied to real-life systems. For example, one can imagine a telecommunications router that computes IPA derivatives, which are updated at packet arrival times. For an on-line management and control application, one may try to use the observed values of *performance metrics and their IPA derivatives* as a component of a control policy that adjusts network parameters, such as source arrival rate (e.g., for access control), link buffer size (e.g., buffer size allocation), and link service rate (e.g., bandwidth allocation). Unfortunately, IPA in traditional queueing systems is often flawed, as it was shown that the *discontinuous* nature of their sample paths tends to render their IPA gradient estimators *biased* in the majority of cases [11,12]. Consequently, attention has recently shifted to fluid queueing systems [14,15], whose *continuous* sample paths give rise to *unbiased* IPA gradients [16–19].

In order to take advantage of the unbiased and nonparametric nature of IPA in fluid-flow queueing systems, this paper advocates an approach where *IPA gradients are computed from data observed in sample paths of essentially discrete queueing systems, but using formulas derived in a continuous setting*. Indeed, the vast majority of queueing models in the literature are described and treated in a discrete setting, so the switch from the discrete paradigm to the continuous one requires justification. To this end, we shall sketch a simple argument that both paradigms can abstract reality at a similar level. More precisely, we argue that both give rise to compatible models, in the sense that discrete and continuous model versions of the same queueing system would have similar performance metrics. As an illustration, consider the sample paths of buffer contents in a single FIFO queue, and sketch how compatible models of a discrete queue and fluid queue with piecewise-constant arrival and service rates might be constructed. In the familiar discrete single queue, these sample paths form step functions, while in its fluid counterpart they are

continuous and piecewise linear. It is clear that the up/down jumps in buffer contents in the former can be matched by ascending/descending ramps in the latter, while the remainder of the paths should be left constant in both cases. For the two models to be compatible, it is necessary to preserve certain queueing invariants. The key invariant is the fact that the service time of equivalent workloads in compatible queues takes the same amount of time. Accordingly, a workload unit (job, packet, etc.) in a discrete queueing model is mapped to a parcel of fluid of equal magnitude, and the inter-arrival and service times in a discrete queue are mapped to arrival and service rates in the fluid counterpart, such that the arrival and departure time points of workload units are synchronized. Note, that any jump can be approximated by a ramp arbitrarily closely by increasing the ramp's slope. Conversely, the termination point of a ramp can be synchronized with a jump. In summary, compatible pairs of discrete and fluid queueing models can be constructed to approximate each other so as to preserve key workload, buffer, and flow invariants. We mention, however, that while both modeling paradigms abstract reality, a fluid queueing model can often capture reality even better than a discrete counterpart. For example, in telecommunications networks, the bits of a transmitted packet actually arrive gradually (at the transmission rate) rather than all at once. In a similar vein, supply chain merchandise takes time to load and unload at arrival and departure points. The point is that the modeler should be free to choose a suitable modeling paradigm. In particular, we may elect to employ fluid queueing models over discrete ones in order to take advantage of the associated unbiased IPA derivatives, thereby circumventing their absence in discrete queueing models.

Several of the aforementioned references contain examples of simple optimization problems where IPA gradients are applied to packet-based models, even as their derivation assumes a fluid-flow setting. The results indicate convergence to the respective optima, thus raising the following question: *Can IPA formulas obtained in a fluid-flow queueing setting be reproduced, or adequately approximated, in the corresponding discrete settings*? An affirmative answer to the above question may render the IPA technique applicable to a significant class of network control and management problems. Recall that this approach requires us to map a discrete flow (e.g., packets) to a corresponding continuous-flow (fluid) counterpart, and such corresponding flows will be referred to as ***compatible flow versions***. Thus, a queueing system may consist entirely of discrete flows (***pure-packet*** model), entirely of fluid flows (***pure-fluid*** model) or a mixture thereof (***hybrid*** model). Finally, any set of queueing models (pure-packet, pure-fluid, or hybrid) consisting entirely of compatible flows will be referred to as *compatible model versions*. Note that in any comparison of compatible model versions, the *baseline model is always the pure-fluid version*, since it is the only one for which IPA derivatives are *unbiased*. In this paper we are interested in packet-based telecommunications models, whose description is available in terms of discrete queueing systems, thereby requiring us to map a discrete flow (e.g., packets) to compatible fluid or hybrid counterparts.

In a previous paper [20], we described a software environment, dubbed ***Hybrid Network Simulator*** (*HNS*), which provides a discrete-event simulation testbed for experimentation with compatible queueing networks (in particular, telecommunications networks). Fluid flows are embedded in the discrete-event framework by introducing flow-rate-change (jump) events in rate functions that are piecewise-constant in time. HNS is designed to facilitate a fundamental modeling tradeoff between simulation running time and accuracy. Modern high-bandwidth telecommunications networks are a case in point. Simulated at the packet level, accurate models of such networks can require prohibitively high CPU times in consequence of the enormous numbers of packets processed by them. In contrast, a compatible fluid-flow simulation is less accurate, but can run much faster than its discrete-flow counterpart (speedups of some three orders of magnitude have been reported in [20]). HNS seamlessly integrates the packet (discrete) and fluid (continuous) paradigms into a hybrid simulation paradigm. Modelers can explore the speed/accuracy tradeoff, since jobs (here, messages) arrive at compatible networks in identical (discrete) arrival processes, but the user can freely select the transport mode (packets or fluid), and HNS generates the requisite compatible flows (variance reduction is afforded by using the same random number streams). IPA derivative computation is incorporated into HNS for all three types of compatible model versions (pure-packet, pure-fluid, or hybrid). These are based on formulas derived in fluid-flow setting in [16] and [17].

This paper studies IPA derivatives computed from a suite of compatible model versions in telecommunications settings. It compares the accuracy of the IPA long-run derivatives across versions, and studies the stabilization of their values under various stopping rules. The results lend empirical support to the contention that IPA derivatives can be accurately computed from discrete versions by adopting a fluid-flow view. Furthermore, as suggested by the functional form of the IPA derivatives, their long-run values stabilize quite fast. The results reported here summarize the work in [21].

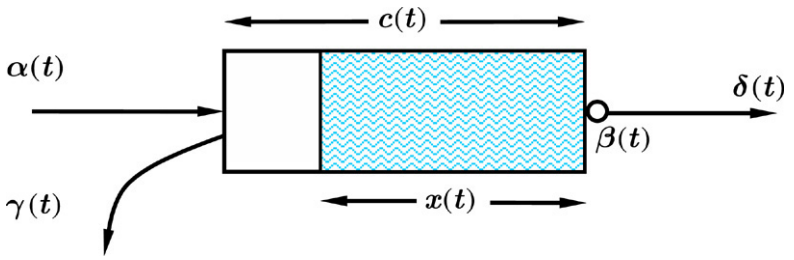

Fig. 1. The basic CFM.

The rest of the paper is organized as follows. Section 2 presents a mathematical description of the fluid-flow FIFO single-queue model employed in this paper. Section 3 provides a brief overview of IPA gradient formulas for this queueing model, while Section 4 describes briefly their implementation in HNS. Section 5 studies and compares the accuracy and stabilization of IPA gradients across compatible model versions of the queueing model under study. Finally, Section 6 concludes the paper.

2. The basic CFM

The fluid queueing systems studied in this paper belong to the basic ***continuous flow model*** (*CFM*) class, also called the ***stochastic flow model*** (*SFM*) class [14,16]. This class is obtained as a straightforward mapping of single discrete queues to single fluid queues (see Fig. 1).

Mathematically, a CFM is determined by a given set of right-continuous stochastic processes, referred to as ***defining processes*** and defined over a common probability space, as follows.

- $\{\alpha(t)\}$ is the inflow rate process of fluid into the CFM, where $\alpha(t)$ is the fluid arrival rate at the system at time t.
- $\{\beta(t)\}$ is the service rate process of fluid in the CFM, where $\beta(t)$ is the fluid service rate at time t.
- $\{c(t)\}$ is the buffer capacity (buffer size) process (often a fixed constant), where $c(t)$ is the buffer capacity at time t.

The defining processes above determine all other stochastic processes of interest, referred to as ***derived processes***. These include the following processes.

- $\{x(t)\}$ is the buffer workload (occupancy) process, where $x(t)$ is the fluid volume in the CFM buffer at time t, governed by the stochastic differential equation

$$\frac{d}{dt^+}x(t) = \begin{cases} 0, & \text{if } x(t) = 0 \quad \text{and} \quad \alpha(t) - \beta(t) \le 0 \\ 0, & \text{if } x(t) = c(t) \quad \text{and} \quad \alpha(t) - \beta(t) \ge 0 \\ \alpha(t) - \beta(t), & \text{otherwise} \end{cases}$$

- $\{\delta(t)\}$ is the fluid discharge (outflow) rate process from the CFM, where $\delta(t)$ is the departure rate of fluid at time t, given by

$$\delta(t) = \begin{cases} \beta(t), & \text{if } x(t) > 0 \\ \alpha(t), & \text{if } x(t) = 0 \end{cases}$$

- $\{\gamma(t)\}$ is the loss (overflow) rate process from the CFM, where $\gamma(t)$ is the rate of lost fluid at time t, given by

$$\gamma(t) = \begin{cases} \max\{\alpha(t) - \beta(t), 0\}, & \text{if } x(t) = c(t) \\ 0, & \text{if } x(t) < c(t). \end{cases}$$

The computation of IPA gradient formulas from CFM sample paths in the next section requires the partitioning of the simulation horizon, $[0, T]$. The resulting partition consists of three types of period, in accordance with the state of the buffer, as follows.

- An ***empty*** period is a closed extremal interval during which the buffer is empty.
- A ***partial*** period is an open extremal interval during which the buffer is neither full nor empty.
- A ***full*** period is a closed extremal interval during which the buffer is full.

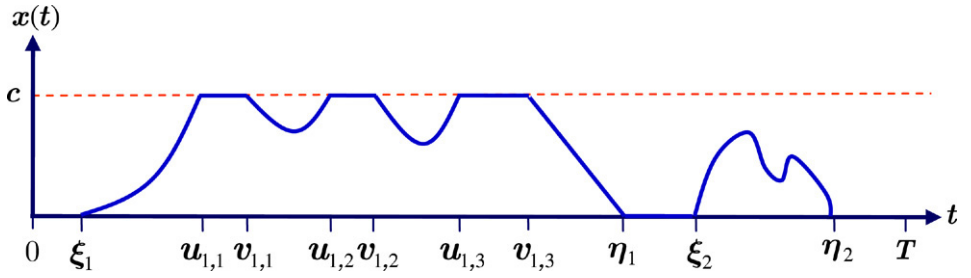

Fig. 2. Partitioning a time horizon.

By an ***extremal interval*** we mean an interval with end points obtained via the inf and sup operations, respectively. We refer to partial and full periods as ***buffering (non-empty) periods***, and refer to empty and partial periods as ***non-full periods***. Furthermore, a ***lossy buffering period*** is one that experienced some loss.

Fig. 2 illustrates such a partition of the time horizon, where

- $[0, \xi_1]$, $[\eta_1, \xi_2]$ and $[\eta_2, T]$ are empty periods;
- $[\xi_1, u_{1,1}]$, $[v_{1,1}, u_{1,2}]$, $[v_{1,2}, u_{1,3}]$, $[v_{1,3}, \eta_1]$ and $[\xi_2, \eta_2]$ are partial periods;
- $[u_{1,1}, v_{1,1}]$, $[u_{1,2}, v_{1,2}]$ and $[u_{1,3}, v_{1,3}]$ are full periods.

Here, we assume constant buffer capacity, i.e., $c(t) = c$, and omit the parameter θ in the variables demarcating period boundaries on the horizontal axis (see the next section).

We are concerned with the following performance metrics over $[0, T]$.

- Loss rate: $L_V = \frac{1}{T}\int_0^T \gamma(t)\,dt$.
- Time average of buffer workload: $L_W = \frac{1}{T}\int_0^T x(t)\,dt$.

The IPA derivatives of these metrics will be exhibited in the next section.

3. IPA gradients for the basic CFM

Suppose that the CFM under consideration depends on some real-valued parameter θ, assumed to belong to a compact (bounded and closed) interval, Θ. For instance, θ might be a parameter related to the buffer size, service rate, or inflow rate.

Explicit formulas of various IPA derivatives in an open-loop CFM setting (i.e., without feedback flows) have been derived in a number of papers (e.g., [16,17]) using sample path analysis. These papers have shown various IPA gradients to be unbiased and fast to compute. Furthermore, by virtue of the sample path analysis, these IPA gradients are nonparametric in the sense that they can be computed from data without any knowledge of the underlying probability law. Recall that these properties hold out the promise of utilizing IPA gradient estimates as an ingredient of on-line management and control of telecommunications networks.

Before exhibiting the formulas for the aforementioned IPA derivatives, we introduce some notation. Let $[0, T]$ be the time horizon; however, the dependence on T will be suppressed to simplify the notation. Let the buffering periods in $[0, T]$ be denoted by

$$B_k(\theta) = (\xi_k(\theta), \eta_k(\theta)), \quad k = 1, \ldots, K(\theta),$$

for some $K(\theta) > 0$, where $\xi_k(\theta)$ and $\eta_k(\theta)$ are the start time and end time of $B_k(\theta)$, respectively. Buffering periods are classified according to whether or not they experienced some loss; the former are called ***lossy buffering periods***, and the latter ***non-lossy buffering periods***. Define the index set

$$\Phi(\theta) = \{1 \le k \le K(\theta): \text{some loss occurred during } B_k(\theta)\},$$

and let

$$N_T(\theta) = |\Phi(\theta)|,$$

where vertical bars denote cardinality.

For each $k \in \Phi(\theta)$, let $M_k(\theta)$ be the number of full periods in $B_k(\theta)$, and denote the associated full periods in increasing order by

$$F_{k,m}(\theta) = [u_{k,m}(\theta), v_{k,m}(\theta)], \quad m = 1, \ldots, M_k(\theta),$$

where $u_{k,m}(\theta)$ and $v_{k,m}(\theta)$ denote, respectively, the start time and end time of $F_{k,m}(\theta)$. Finally, for a given $\theta \in \Theta$, we define a ***sample path event*** as an occurrence in a sample path at a particular time point. The computation of IPA derivatives in this section makes use of two types of sample path event:

- an ***exogenous sample path event*** is a jump in either $\{\alpha(\theta; t)\}$ or $\{\beta(\theta; t)\}$;
- an ***endogenous sample path event*** corresponds to the buffer becoming either empty or full.

We now proceed to exhibit the IPA derivatives of interest. We shall assume that $\theta \in \Theta$ for a given closed and bounded interval Θ whose left-hand point is positive, and that for every $\theta \in \Theta$, the following hold.

- With probability 1, the process $\{\alpha(\theta, t) - \beta(\theta, t)\}$ is piecewise continuously-differentiable in t in the interval $[0, T]$. In particular, this condition holds for piecewise-constant inflow-rate and service-rate processes.
- With probability 1, no multiple sample path events occur simultaneously.
- The sample derivatives $L'_V(\theta)$ and $L'_W(\theta)$ exist, with probability 1.

The formulas in Propositions 1 through 6 to follow were derived in [16], and are valid (and unbiased) for a basic CFM when the initial state of a sample path consists of an empty buffer (cold start).

3.1. IPA derivatives with respect to buffer size

Throughout this subsection, the parameter θ is the buffer size, i.e., $c(\theta) = c$, and we make the assumption that the processes $\{\alpha(t)\}$ and $\{\beta(t)\}$ are independent of θ.

Proposition 1 (*Cf. [16, Proposition 3.1]*). *For every $\theta \in \Theta$,*

$$L'_V(\theta) = -\frac{1}{T} N_T(\theta). \tag{1}$$

In words, the IPA derivative is the ratio of the negative number of lossy buffering periods in the time horizon, $[0, T]$*, to the time horizon length, T.*

Proposition 2 (*Cf. [16, Proposition 3.2]*). *For every $\theta \in \Theta$,*

$$L'_W(\theta) = \frac{1}{T} \sum_{k \in \Phi(\theta)} [\eta_k(\theta) - u_{k,1}(\theta)]. \tag{2}$$

In words, only lossy buffering periods, $B_k(\theta)$, contribute to the IPA derivative, and the corresponding addend in the sum is the length of the interval extending from the start point of the first full period in $B_k(\theta)$ to the end point of $B_k(\theta)$.

3.2. IPA derivatives with respect to a service rate parameter

Throughout this subsection, the parameter θ is a parameter of the service rate process, $\{\beta(\theta; t)\}$, and we make the following assumptions.

- The inflow process, $\{\alpha(t)\}$, and the buffer size, c, are independent of θ.
- For all $\theta \in \Theta$ and for all $t \in [0, T]$,

$$\frac{\partial}{\partial \theta} \beta(\theta; t) = \beta'(\theta; t) = 1.$$

Proposition 3 (*Cf. [16, Proposition 4.1]*). *For every $\theta \in \Theta$,*

$$L'_V(\theta) = -\frac{1}{T} \sum_{k \in \Phi(\theta)} [v_{k,M_k(\theta)}(\theta) - \xi_k(\theta)]. \tag{3}$$

In words, only lossy buffering periods, $B_k(\theta)$, contribute to the IPA derivative, and the corresponding addend in the sum is the length of the interval extending from the start point of $B_k(\theta)$ to the end point of the last full period in $B_k(\theta)$.

Proposition 4 (*Cf. [16, Proposition 4.2]*). *For every* $\theta \in \Theta$,

$$L'_W(\theta) = -\frac{1}{2T} \sum_{k=1}^{K(\theta)} \sum_{m=1}^{M_k(\theta)+1} [u_{k,m}(\theta) - v_{k,m-1}(\theta)]^2. \tag{4}$$

In words, only buffering periods, $B_k(\theta)$, *contribute to the IPA derivative, and the corresponding addend in the sum is the sum of squares of the lengths of the partial periods in* $B_k(\theta)$.

3.3. IPA derivatives with respect to an inflow rate parameter

In this subsection, the parameter θ is a parameter of the inflow rate processes $\{\alpha(\theta; t)\}$, and $\theta \in \Theta$ for a given closed and bounded interval Θ. Assume further that for all $\theta \in \Theta$ and for all $t \in [0, T]$,

$$\frac{\partial}{\partial\theta}\alpha(\theta; t) = \alpha'(\theta; t) = \begin{cases} 1, & \text{if } \alpha(\theta; t) > 0 \\ 0, & \text{if } \alpha(\theta; t) = 0. \end{cases}$$

Throughout this subsection, we make the following assumptions.

(a) The service process, $\{\beta(t)\}$, and the buffer size, c, are independent of θ.
(b) The jump points of $\alpha(\theta; t)$ do not depend on θ.
(c) For every $t \in [0, T]$, the function $\alpha(\theta; t)$ is continuously differentiable in θ.
(d) There exists $K(\theta) < \infty$ such that with probability 1,

$$\sup\{|\alpha'(\theta; t)| : \theta \in \Theta; t \in [0, T]\} \le K(\theta).$$

Proposition 5 (*Cf. [16, Proposition 5.1]*). *For every* $\theta \in \Theta$,

$$L'_V(\theta) = \frac{1}{T} \sum_{k \in \Phi(\theta)} \int_{\xi_k(\theta)}^{v_{k,M_k(\theta)}(\theta)} \alpha'(\theta; \tau)\, d t. \tag{5}$$

In words, only lossy buffering periods, $B_k(\theta)$, *contribute to the IPA derivative, and the contribution of each addend in the sum is the total length of intervals with positive inflow rates lying between the start point of* $B_k(\theta)$ *and the end point of the last full period in* $B_k(\theta)$.

Proposition 6 (*Cf. [16, Proposition 5.2]*). *For every* $\theta \in \Theta$,

$$L'_W(\theta) = \frac{1}{T} \sum_{k=1}^{K(\theta)} \sum_{m=1}^{M_k(\theta)+1} \int_{v_{k,m-1}(\theta)}^{u_{k,m}(\theta)} \int_{v_{k,m-1}(\theta)}^{t} \alpha'(\theta; \tau)\, d\tau\, dt. \tag{6}$$

4. Implementation of IPA gradients in HNS

The IPA formulas above were derived for the pure-fluid paradigm. However, as explained in Section 1, the implementation of IPA gradients in discrete-event simulation requires algorithms for their computation not only for the pure-fluid setting, but also in compatible pure-packet and hybrid settings. In this section we briefly describe the implementation of IPA gradient computations for single queues in the HNS application [20].

First, we describe briefly the modeling worldview of HNS (see [20] for more details). In HNS, a ***network*** is a (fixed) directed graph consisting of ***nodes*** and ***links***, where a node represents a network location and a link connects a pair of nodes. Each link houses a service facility (a shared server with a prescribed service speed and a shared buffer with a prescribed size) for processing (serving) workload carried by transactions. ***Workload*** can be ***discrete*** (e.g., packets, jobs, etc., as in traditional queueing theory) or ***continuous*** (fluid). While discrete workload is represented in HNS in the standard way, continuous workload is represented by piecewise-constant flow rates (both arrival rates and service rates), and their associated durations. This representation of rate functions in time is adequately general (any reasonable rate function can be approximated arbitrarily closely by a piecewise-constant function), and simplifies the evaluation of time-average integrals (the integrands are piecewise-linear). Transaction arrivals at the network and departures from it are modeled identically for discrete and continuous workload, in the traditional way.

The network interacts with an exogenous environment consisting of sources and sinks, which are attached to nodes. A ***source*** is an ingress point for workload to enter the network, while ***a sink*** is an egress point for workload to depart

from the network. A source generates a transaction stream, according to the interarrival process, and has the following attributes:

1. an arrival process of transactions and their associated workload distribution;
2. an injection process (piecewise-constant stochastic process of injection rates with random durations);
3. a workload type (discrete or continuous);
4. a priority (to resolve contention for buffer and server resources);
5. an itinerary (path through the network starting in a source and terminating in a sink);
6. a protocol (rules that govern the transport of workload, such as flow control);
7. a transport mode (discrete or continuous).

Once generated, each transaction's workload is either ***packetized*** or ***fluidized***, depending on the user's choice of the transport mode. The attendant workload is injected from the source into the network, and then traverses the network along its itinerary. It contends for server and buffer resources at each link according to its priority, and finally departs from the network at its itinerary's sink.

In HNS, a link can be in one of the following states at any given time: **IDLE-EMPTY**, **BUSY-EMPTY**, **BUSY-PARTIAL**, and **BUSY-FULL**. These states are used to manage the workload admission logic of flows at link buffers. The HNS ***parceling algorithm*** seamlessly integrates the processing of workload flows at link buffers for both discrete and continuous transactions. Flow workloads at a link buffer are organized in *multiparcels*, each being a vector of volumes of multiple incoming workloads into the buffer, such that the incoming rate of each constituent flow is constant (if the incoming rate of a flow changes, the current multiparcel is closed, and a new mulitparcel is opened for the new vector of constant rates). Multiparcels are created by the flows' ***effective inflow rates*** (rates at which the flow actually enters the current link buffer), which is computed based on the flows' ***offered arrival rates*** (rates at which the flow leaves the upstream link or source) and the current link's state and service rate.

As it turns out, the main task in computing IPA sample gradients is the identification of the time boundaries of various queue states. In HNS, these are the time points at which the link state changes. In a pure-fluid setting, these states can be identified unambiguously in a natural way. However, whenever packets are involved (pure-packet setting or hybrid setting) some conceptual problems arise in identifying the packet counterparts of the states **BUSY-EMPTY** and **BUSY-FULL**. The problems involved and their solutions will be briefly explained next.

4.1. Time boundaries between empty and non-empty periods

Consider a packet being routed from some origination link to some destination link. Recall that in this case, the entire packet workload is routed in zero time (in contrast, a fluid workload is routed over a period of time). When contemplating the mapping of a packet routing to the fluid-flow setting, the following discordance emerges. Focusing on the destination link, it is clear that packet routing would cause its state to alternate between empty and nonempty states. However, its fluid-flow counterpart may behave differently in light traffic, since a positive fluid flow can occur in a continually empty buffer without changing its state (this happens in an empty buffer when the arrival rate does not exceed the service rate). The result is that the IPA derivatives computed for compatible fluidized and packetized flows can diverge, so a method to harmonize the computation in such scenarios is needed. What is needed conceptually is a way to "fluidize" packet behavior so as to remove the state transitions inherent in packet flows under light traffic, but only for the purpose of mapping the computation of IPA derivatives from the fluid paradigm to the packet or hybrid paradigms.

To this end, HNS associates with a packet in service a so-called ***nominal arrival rate***. This rate is defined as either the ***effective injection rate*** (in cases where the packet is injected into the network from a source) or the packet's ***allocated service rate*** at the origination link (in all other cases). As soon as the packet enters service in the origination link, the packet's nominal arrival rate is incorporated into the inflow rates into the destination link, and that nominal arrival rate is used to determine the buffer's state transitions.

4.2. Time boundaries between full and non-full periods

In a hybrid network, a link buffer is considered full in either of the following two scenarios.

1. A fluid "buffer-full" event occurs.
2. A packet is discarded because the destination buffer cannot accommodate the packet's workload.

The latter case causes the following discordance when contemplating the mapping of a packet routing to the fluid-flow setting. Upon the arrival of the packet at the destination buffer, the packet is discarded (due to insufficient buffer capacity) and the buffer state is declared as "full". However, the moment the packet is discarded, the buffer state changed back to "non-full", since that buffer can still have some residual capacity. Hence, the buffer state would undergo two instantaneous state transitions. However, in pure-fluid networks, this kind of state transition is absent.

In order to compute conceptually compatible IPA derivatives for packet and hybrid streams, it is essential to harmonize the fluid and packet worldviews for the second scenario above. To this end, HNS again uses the notion of packet nominal arrival rate, and harmonizes the packet and fluid worldviews by distinguishing between two cases.

(1) If $\alpha(t) > \beta(t)$, where $\alpha(t)$ includes all relevant packet nominal arrival rates, then HNS glosses over the instantaneous state transition to a "non-full" state, and simply declares the link buffer to stay in a "full" state.
(2) Otherwise, if $\alpha(t) \leq \beta(t)$, then HNS declares the buffer to have undergone a state transition from a "full" state to a "non-full" state.

5. Experimentation with IPA using HNS

In this section, we describe HNS simulation experiments with IPA computations, summarizing the work in [21]. Recall that the closed form formulas for IPA derivatives in Section 3 are derived for *open-loop* flows, i.e., network flows with no feedback loops. In particular, the UDP transport protocol [5] implemented in HNS meets this requirement and, consequently, the experimentation in this section studies telecommunications networks with UDP-type streams only.

The goal of this section is to study statistically the six IPA derivatives of Section 3, collected from compatible simulation model versions (pure-packet, pure-fluid, and hybrid versions) in various network models.

1. Loss rate as a function of buffer size (**IPA-1**).
2. Workload time average as a function of buffer size (**IPA-2**).
3. Loss rate as a function of a service rate parameter (**IPA-3**).
4. Workload time average as a function of a service rate parameter (**IPA-4**).
5. Loss rate as a function of an arrival rate parameter (**IPA-5**).
6. Workload time average as a function of an arrival rate parameter (**IPA-6**).

Recall that the IPA derivative formulas of Section 3 are only valid when the initial state of a sample path consists of an empty buffer (cold start). Consequently, all replications in this section have no warm-up period to ensure IPA computations from a cold start.

Since all the IPA derivatives above are time averages of the form

$$G(T) = \frac{C(T)}{T}, \tag{7}$$

where $C(T)$ is monotone non-decreasing, it is reasonable to expect them to stabilize (approximately converge) after a sufficiently long simulation horizon of minimal length, $T_{\min}$. More precisely, when the system is ***ergodic***, then the long-run value of a (time average) IPA derivative coincides with its invariant (equilibrium) mean [22, Chapter 6]. From now on, allusions to estimating the stable values of IPA derivatives will refer to the estimation of the aforementioned invariant mean.

As will be seen, experimentation with IPA derivatives in HNS supports this expectation. Consequently, in order to estimate an IPA derivative from a replication, it is necessary to devise a good stopping rule which can be employed to detect approximate convergence. Since detecting the stopping point in time is computationally intensive, we considered the following two computationally efficient stopping rules.

HSR: This is a heuristic stopping rule based on inspection of iterative pilot runs and requires human interaction. A heuristic simulation horizon, T_H, is selected conservatively, such that all the IPA derivatives appear to stabilize "sufficiently". IPA derivatives obtained via ***HSR*** will be referred to as ***almost stable***.

MSR: This is a two-stage mixed (heuristic and algorithmic) stopping rule, computed from a replication per IPA derivative as follows.

Stage 1: Determine a minimal time length, $T_{\min}$, applicable to all IPA derivatives by heuristic experimentation with pilot runs, in the vein of ***HSR***. The rationale for $T_{\min}$ is the need to add a measure of stabilization to the IPA derivatives and prevent premature stopping of their computation.

Stage 2: Once $T_{\min}$ elapses, this stage starts to test for convergence. To economize on the computational effort involved, it only tests every $\frac{N}{100}$ IPA derivative updates, where N is the total number of IPA derivative updates observed during $[0, T_{\min}]$. Let t_k, $k = 0, 1, 2, \ldots$, denote the simulation time at which the k-th test occurs. The iteration stops as soon as the relative deviation of two successive IPA derivative observations, $G(t_{k-1})$ and $G(t_k)$, falls below $\varepsilon = 0.005$, namely, when

$$\left|\frac{G(t_k) - G(t_{k-1})}{G(t_{k-1})}\right| < \varepsilon. \tag{8}$$

IPA derivatives obtained via *MSR* will be referred to as ***approximately stable***.

The following subsections describe four studies.

1. ***HSR accuracy study***: compares the relative deviations of almost stable IPA derivatives across simulation model versions, based on one long replication subject to stopping rule ***HSR***.
2. ***MSR convergence study***: studies the convergence of approximately stable IPA derivatives as well as stopping time values, based on multiple replications subject to stopping rule ***MSR***.
3. ***MSR accuracy study***: compares the relative deviations of various statistics of IPA derivatives across simulation model versions, based on multiple replications subject to stopping rule ***MSR***.
4. ***MSR-to-HSR comparison study***: compares the relative deviations of corresponding almost stable and approximately stable IPA derivatives within versions, based on one compatible replication subject to their respective stopping rule.

The accuracy measure used in studies 1, 3, and 4 above utilizes the relative deviation

$$D(statistic) = \frac{statistic - baseline}{baseline} \times 100\%. \tag{9}$$

Here, *baseline* refers to an IPA derivative value in a pure-fluid version (baseline model), while *statistic* refers to its counterpart in a pure-packet or hybrid version. The tables to follow display the relative deviation values enclosed in parentheses under the second-line heading "% deviation". The simulation model versions were made compatible in that they have identical network configurations, as well as the same arrival processes at network sources and service processes at network links. Thus, the versions are subjected to identical loads and only differ in the transport mode of the workload.

5.1. HSR accuracy study

The goal of this study is to gauge the relative accuracy of almost stable IPA derivatives across simulation model versions. For each version (pure-fluid, pure-packet, and hybrid), this study runs one long replication of length T_H and computes all IPA derivatives. A suite of two network models were studied:

1. a simple 5-source, 4-link network;
2. a 12-node, 11-link network with a bottleneck link.

The next two subsections describe the comparison study for each of the models above.

5.1.1. A simple 5-source, 4-link network

Fig. 3 depicts a simple network with 5 nodes (empty circles), 4 links (arrows), 4 sources (filled trapezoids), and 4 sinks (filled circles). Each source generates one UDP flow, whose itinerary (path in the network) is texture coded. Network model parameters are as follows.

- Incoming UDP message workloads are infinite (so only one message arrives at each UDP source and generates the entire workload).
- Each UDP source injection rate is an ON/OFF process with iid exponential durations of mean 1 s and an ON rate of 10 Mbps.

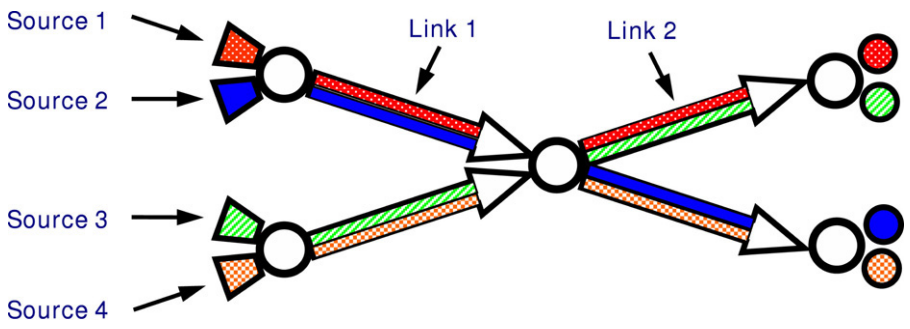

Fig. 3. A simple 5-node, 4-link network.

Table 1
Comparison of IPA statistics, subject to *HSR*, for Link 1 of the simple network

IPA derivative	Pure fluid	Pure packet (% deviation)	Hybrid (% deviation)
Loss rate as a function of buffer size	−0.01907	−0.01904 (−0.131%)	− 0.01907 (0%)
Workload time average as a function of buffer size	0.05701	0.05003 (−12.244%)	0.05702 (0.009%)
Loss rate as a function of a service rate parameter	−0.06225	−0.06226 (0.018%)	−0.06222 (−0.040%)
Workload time average as a function of a service rate parameter	−0.46144	−0.48566 (5.249%)	−0.46148 (0.009%)
Loss rate as a function of an arrival rate parameter	0.06209	0.06210 (0.022%)	0.06021 (−3.029%)
Workload time average as a function of an arrival rate parameter	0.45877	0.43319 (−5.575%)	0.45210 (−1.453%)

- UDP packets are of size 8 Kb.
- All link rates are 18 Mbps.
- All links have buffer size 10 Mb.
- In the hybrid simulation version, Source 1 and Source 4 inject packet streams, while Source 2 and Source 3 inject fluid streams. Therefore, each link carries one packet stream and one fluid stream.

In view of the symmetry of this network model, we collected the IPA statistics only for Links 1 and 2 (see Fig. 3) over the same simulation interval, [0, 600]. Table 1 compares the IPA derivatives, IPA-1 through IPA-6, of the three compatible simulation model versions at Link 1. Table 2 compares the IPA statistics, IPA-1 through IPA-6, of the three simulation model versions at Link 2.

5.1.2. A 12-node, 11-link network with a bottleneck link

Fig. 4 depicts a 12-node, 11-link network with 5 UDP sources. The network model parameters are as follows.

- Incoming UDP message workloads are infinite (so only one message arrives at each UDP source and generates the entire workload).
- Each UDP source injection rate is an ON/OFF process with iid exponential durations of mean 1 s and an ON rate of 990 Mbps.
- All UDP packets are of size 8 Kb.
- All non-bottleneck links have a transmission rate of 1 Gbps.
- The bottleneck link has a transmission rate of 1.5 Gbps.
- All link buffers have size 30 Mb.

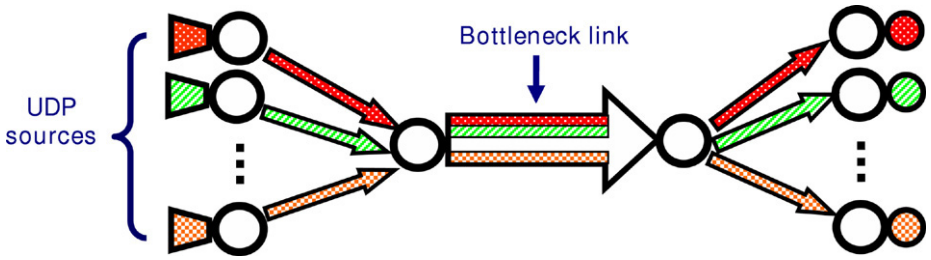

Fig. 4. A network with a bottleneck link.

Table 2
Comparison of IPA statistics, subject to *HSR*, for Link 2 of the simple network

IPA derivative	Pure fluid	Pure packet (% deviation)	Hybrid (% deviation)
Loss rate as a function of buffer size	−0.00935	−0.00935 (−0.003%)	−0.00935 (0.017%)
Workload time average as a function of buffer size	0.01800	0.01877 (4.269%)	0.01775 (−1.371%)
Loss rate as a function of a service rate parameter	−0.02193	−0.02111 (−3.768%)	−0.02063 (−5.935%)
Workload time average as a function of a service rate parameter	−0.41924	−0.41473 (−1.077%)	−0.44128 (5.257%)
Loss rate as a function of an arrival rate parameter	0.02193	0.01922 (−12.340%)	0.01935 (−11.774%)
Workload time average as a function of an arrival rate parameter	0.39881	0.39781 (−0.250%)	0.42225 (5.879%)

Table 3
IPA statistics, subject to *HSR*, for the network with a bottleneck link

IPA derivative	Pure fluid	Pure packet (% deviation)	Hybrid (% deviation)
Loss rate as a function of buffer size	−1.11888	−1.12054 (0.15%)	−1.11888 (0%)
Workload time average as a function of buffer size	0.22905	0.22901 (−0.02%)	0.22905 (0%)
Loss rate as a function of a service rate parameter	−0.21814	−0.21806 (−0.04%)	−0.21843 (0.13%)
Workload time average as a function of a service rate parameter	−0.00660	−0.00659 (−0.04%)	−0.00657 (−0.42%)
Loss rate as a function of an arrival rate parameter	0.21805	0.21706 (−0.45%)	0.21828 (0.11%)
Workload time average as a function of an arrival rate parameter	0.00660	0.00659 (−0.04%)	0.00657 (−0.43%)

- In the hybrid version, one of the UDP sources injects a packet stream, while the other four UDP sources inject fluid streams.

Table 3 compares the IPA derivatives, IPA-1 through IPA-6, at the bottleneck link of the three simulation model versions over the same simulation interval, [0, 600].

The relative deviations of the IPA derivatives in the above tables are mostly in the range of 1–5% (the worst relative deviation is about 12.3%), which indicates that the three different simulation model versions give rise to very close IPA derivative values. The higher accuracy of the IPA derivatives in this model as compared to the previous model is due to the fact that this model has higher multiplexing levels of packet flows and, consequently, the behavior of

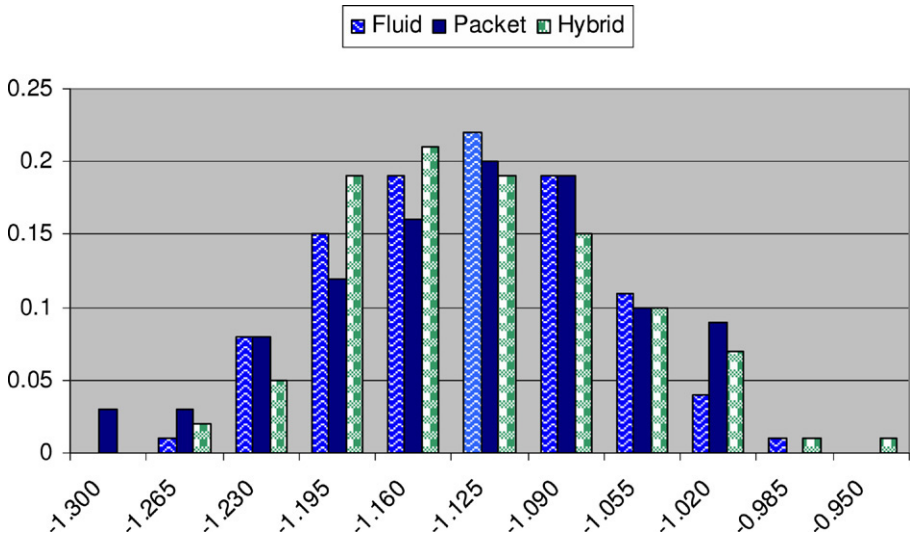

Fig. 5. Histograms of IPA derivatives of loss rate as a function of buffer size.

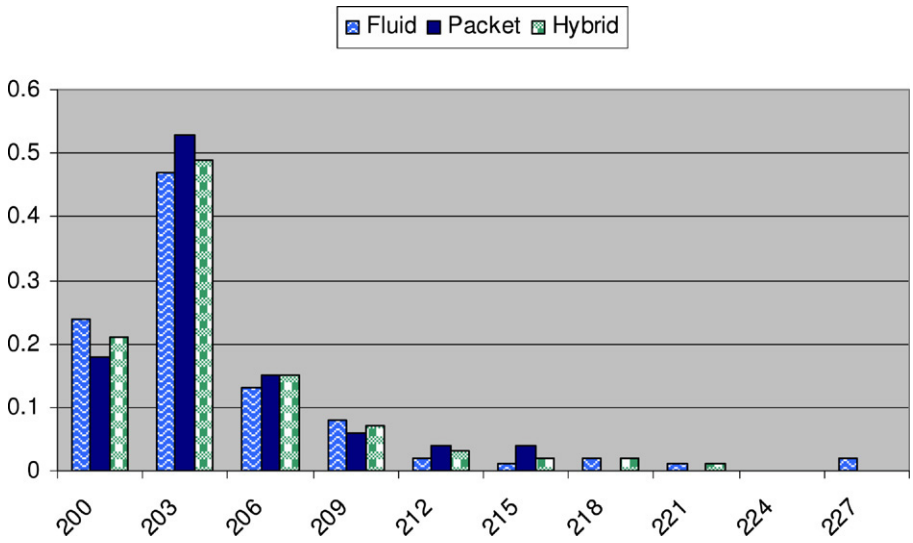

Fig. 6. Histograms of stopping times for IPA derivatives of loss rate as a function of buffer size.

its packet streams is more fluid-like. Overall, this study attests to the efficacy of using fluid-based IPA derivatives as approximations for pure-packet and hybrid models.

5.2. *MSR convergence study*

The goal of this study is to gauge the convergence of the approximately stable IPA derivatives within simulation model versions, subject to stopping rule *MSR*. It also compares the corresponding stopping time values. To this end, 100 replications were run for each simulation model version, and the approximately stable IPA derivatives, as well as corresponding stopping time values, were collected. Histograms of the approximately stable IPA derivatives and stopping time values were generated from the replications.

The network model we studied is the 12-node, 11-link network with a bottleneck link, depicted in Fig. 4. The simulation results for replications with $T_{\min} = 200$ s are summarized in Figs. 5 through 16. Each figure depicts the histogram of an IPA derivative computed from 100 replications followed by the histogram of the corresponding

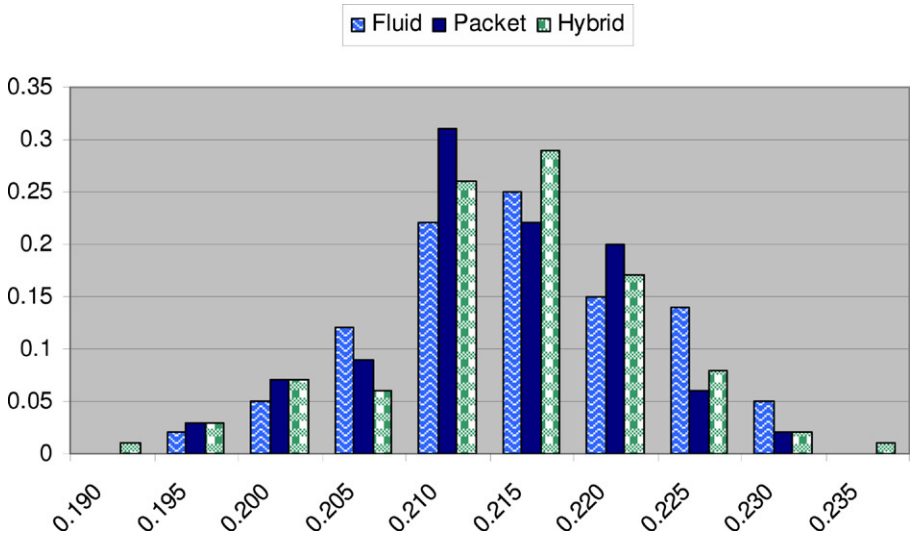

Fig. 7. Histograms of IPA derivatives of workload time average as a function of buffer size.

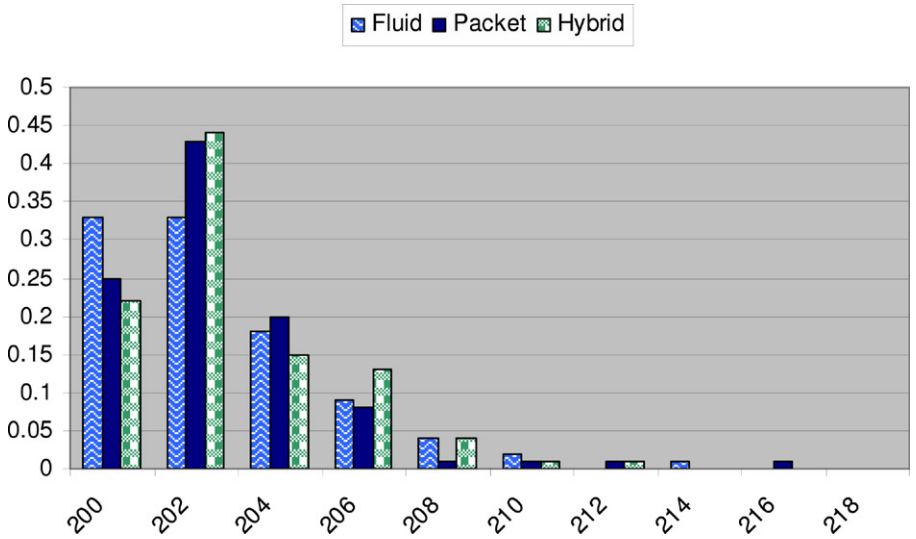

Fig. 8. Histograms of stopping times for IPA derivatives of workload time average as a function of buffer size.

stopping time values. In these figures, statistics for the baseline pure-fluid version are depicted by bars with a wavy texture, those for the pure-packet version are depicted by solid bars, and those for the hybrid version are depicted by bars with a checkered texture.

The figures indicate that convergence to stable IPA derivative values has low variability. To wit, note that all the histograms of approximately stable IPA derivative values are roughly bell-shaped and thin-tailed. Furthermore, their standard deviations (to be displayed in Section 5.3) are quite small relative to the corresponding mean values. Moreover, the stopping-time histograms of all three simulation model versions are pretty close. Interestingly, these histograms are not bell-shaped, but rather are right-skewed.

5.3. MSR accuracy study

The goal of this study is to gauge the relative accuracy of various statistics of approximately stable IPA derivatives across simulation model versions. For each simulation version (pure-fluid, pure-packet, and hybrid), this study ran 100 replications subject to stopping rule *MSR*, with the $T_{\min}$ parameter set to 200 s. Each replication collected all

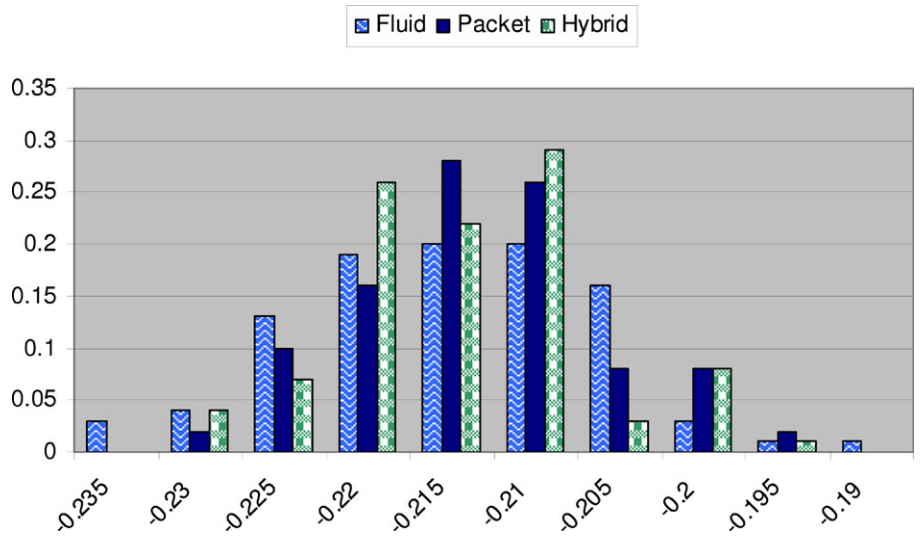

Fig. 9. Histograms of IPA derivatives of loss rate as a function of a service rate parameter.

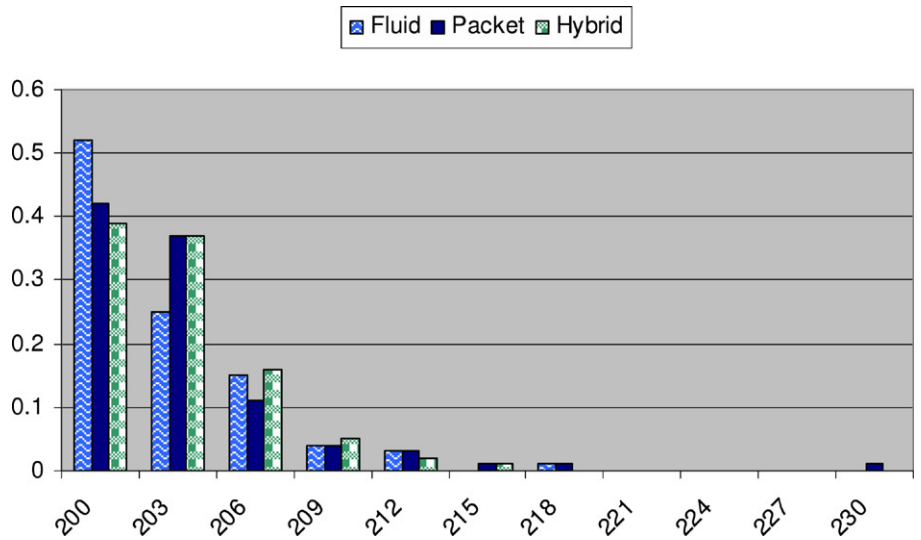

Fig. 10. Histograms of stopping times for IPA derivatives of loss rate as a function of a service rate parameter.

IPA derivatives at the stopping time, as well as the following statistics: minimum, maximum, mean, and standard deviation. These statistics were then compared for relative accuracy.

The network model studied in this section is the 12-node, 11-link network with a bottleneck link, depicted in Fig. 4. The results are summarized in Tables 4 through 9. Tables 4 through 9 show generally good agreement of the statistics of approximately stable IPA derivatives across simulation versions. All statistics other than the standard deviations fall within a few percentage points of each other (in particular, all means are within 1%). However, the coefficients of variations are very close. This study attests to the efficacy of using fluid-based IPA derivatives as approximations for pure-packet and hybrid models, subject to the *MSR* stopping rule.

5.4. MSR-to-HSR comparison study

The goal of this study is to compare compatible almost stable and approximately stable IPA derivatives for each simulation model version, in order to gauge the tradeoff between the accuracy of IPA derivatives subject to the *HSR* stopping rule and potential savings in time complexity afforded by the *MSR* stopping rule. To this end, we compared

Table 4
Statistics of IPA-1 derivatives, subject to *MSR*

Statistic	Pure-fluid	Pure-packet (% deviation)	Hybrid (% deviation)
Minimum	−1.23953	−1.29985 (4.87%)	−1.23847 (−0.09%)
Maximum	−0.96017	−0.99596 (3.73%)	−0.94059 (−2.04%)
Mean	−1.114	−1.11624 (0.20%)	−1.11516 (0.10%)
Standard deviation	0.058955	0.070113 (18.93%)	0.061811 (4.84%)

Table 5
Statistics of IPA-2 derivatives, subject to *MSR*

Statistic	Pure-fluid	Pure-packet (% deviation)	Hybrid (% deviation)
Minimum	0.200865	0.200699 (−0.08%)	0.198163 (−1.35%)
Maximum	0.237344	0.237231 (−0.05%)	0.242321 (2.10%)
Mean	0.221837	0.220336 (−0.68%)	0.220748 (−0.49%)
Standard deviation	0.008131	0.00739 (−9.11%)	0.007979 (−1.87%)

Table 6
Statistics of IPA-3 derivatives, subject to *MSR*

Statistic	Pure-fluid	Pure-packet (% deviation)	Hybrid (% deviation)
Minimum	−0.23248	−0.22841 (−1.75%)	−0.22937 (−1.34%)
Maximum	−0.18992	−0.19438 (2.35%)	−0.19166 (0.92%)
Mean	−0.21285	−0.21107 (−0.84%)	−0.21189 (−0.45%)
Standard deviation	0.00857	0.007238 (−15.54%)	0.007289 (−14.95%)

Table 7
Statistics of IPA-4 derivatives, subject to *MSR*

Statistic	Pure-fluid	Pure-packet) (% deviation)	Hybrid (% deviation)
Minimum	−0.00741	−0.00755 (1.89%)	−0.00783 (5.67%)
Maximum	−0.00568	−0.00557 (−1.94%)	−0.00535 (−5.81%)
Mean	−0.00651	−0.00647 (−0.61%)	−0.00648 (−0.46%)
Standard deviation	0.000338	0.000377 (11.54%)	0.000398 (17.75%)

Table 8
Statistics of IPA-5 derivatives, subject to *MSR*

Statistic	Pure-fluid	Pure-packet (% deviation)	Hybrid (% deviation)
Minimum	0.191327	0.193387 (1.08%)	0.191425 (0.05%)
Maximum	0.231203	0.227628 (−1.55%)	0.229889 (−0.57%)
Mean	0.212309	0.210663 (−0.78%)	0.211316 (−0.47%)
Standard deviation	0.008424	0.007225 (−14.23%)	0.007254 (−13.89%)

Table 9
Statistics of IPA-6 derivatives, subject to *MSR*

Statistic	Pure-fluid	Pure-packet (% deviation)	Hybrid (% deviation)
Minimum	0.005686	0.005564 (−2.15%)	0.005352 (−5.87%)
Maximum	0.007388	0.007576 (2.54%)	0.007835 (6.05%)
Mean	0.006488	0.006461 (−0.42%)	0.006471 (−0.26%)
Standard deviation	0.000341	0.000383 (12.32%)	0.0004 (17.30%)

the IPA derivative values obtained from a replication subject to the *HSR* stopping rule and a compatible replication (using the same random number stream) subject to the *MSR* stopping rule.

The network model studied in this section is the 12-node, 11-link network with a bottleneck link, depicted in Fig. 4. The T_H parameter for stopping rule *HSR* was set to 600 s, while the $T_{\min}$ parameter for stopping rule *MSR* was set to 200 s. The results are summarized in Tables 10 through 12. Here, the second and fourth columns of each table display the stopping times under *HSR* and *MSR*, respectively. Furthermore, the third and fifth columns of each table display the attendant almost stable and approximately stable IPA derivative values, with the relative deviations of the latter from the former shown in parentheses.

Tables 10 through 12 show excellent agreement between almost stable and approximately stable IPA derivatives across stopping rules and across compatible simulated model versions. All IPA derivative values are within about 6% regardless of version (interestingly, the relative deviations are also very close across versions). However, the time complexity of computing IPA derivatives under *MSR* is about one-third of their counterparts under *HSR*. This study attests to the efficacy of using the *MSR* stopping rule to compute approximately stable IPA derivatives instead of almost stable IPA derivatives, but at a fraction of the time complexity.

6. Conclusion

The empirical studies described in this paper can be summarized as follows.

1. Pure-fluid IPA derivatives can be accurately computed from pure-packet and hybrid models. This conclusion is supported by the study in Section 5.1 (under *HSR*) and Section 5.2 (under *MSR*), where various compatible simulation model versions (pure-fluid, pure-packet, and hybrid) gave rise to corresponding IPA derivatives within a few percentage points of relative deviation. Since the IPA derivatives are nonparametric, this opens the door for possible practical applications of IPA derivatives to the optimization and control of real-life systems, in addition to experimentation in simulation models.
2. The *MSR* stopping rule yields approximately stable IPA derivatives with relatively little variability across replications. This conclusion is supported by the study in Section 5.3, and motivates using this rule to approximate stable (long-run) IPA derivatives.

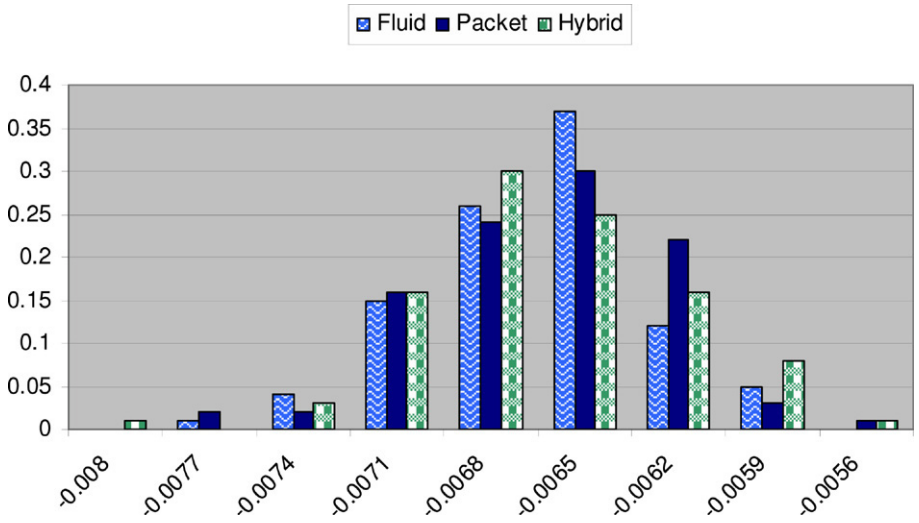

Fig. 11. Histograms of IPA derivatives of workload time average as a function of a service rate parameter.

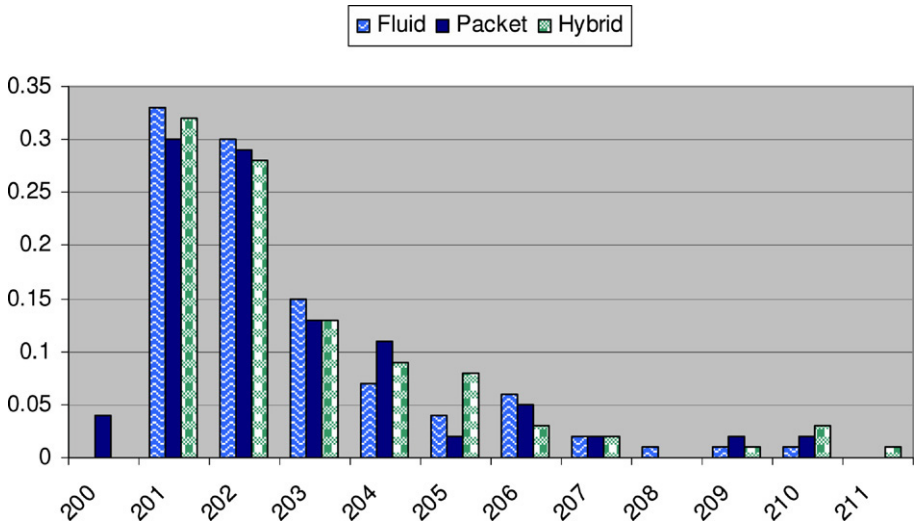

Fig. 12. Histograms of stopping times for IPA derivatives of workload time average as a function of a service rate parameter.

Table 10
Pure-fluid version comparison of IPA derivatives subject to *HSR* and *MSR*

IPA derivative	T_H	HSR	MSR stopping time	MSR (% deviation)
Loss rate as a function of buffer size	600	−1.11888	209.8461	−1.04839 (−6.30%)
Workload time average as a function of buffer size	600	0.22905	201.8987	0.22244 (−2.89%)
Loss rate as a function of a service rate parameter	600	−0.21814	201.5469	−0.20812 (−4.59%)
Workload time average as a function of a service rate parameter	600	−0.00660	201.8987	−0.00676 (2.40%)
Loss rate as a function of an arrival rate parameter	600	0.21805	201.8987	0.20900 (−4.15%)
Workload time average as a function of an arrival rate parameter	600	0.00660	201.8200	0.00675 (2.30%)

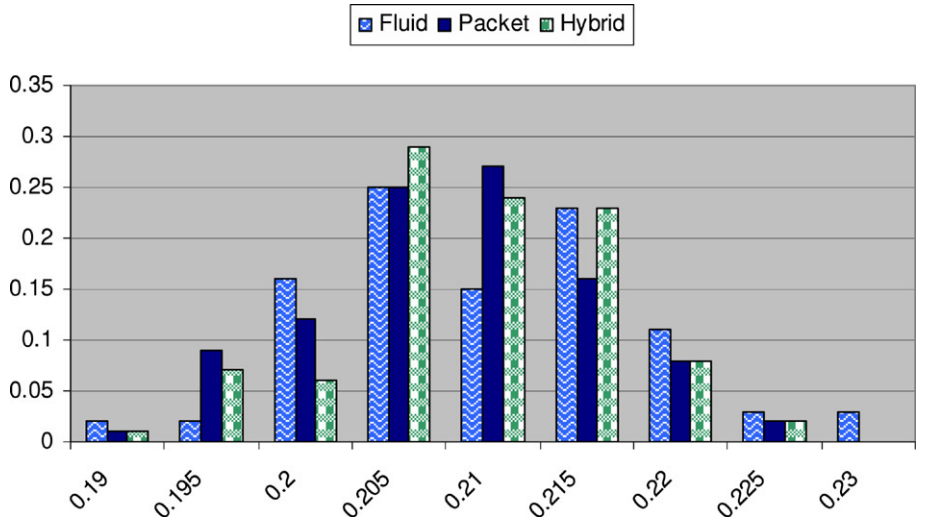

Fig. 13. Histograms of IPA derivatives of loss rate as a function of an arrival rate parameter.

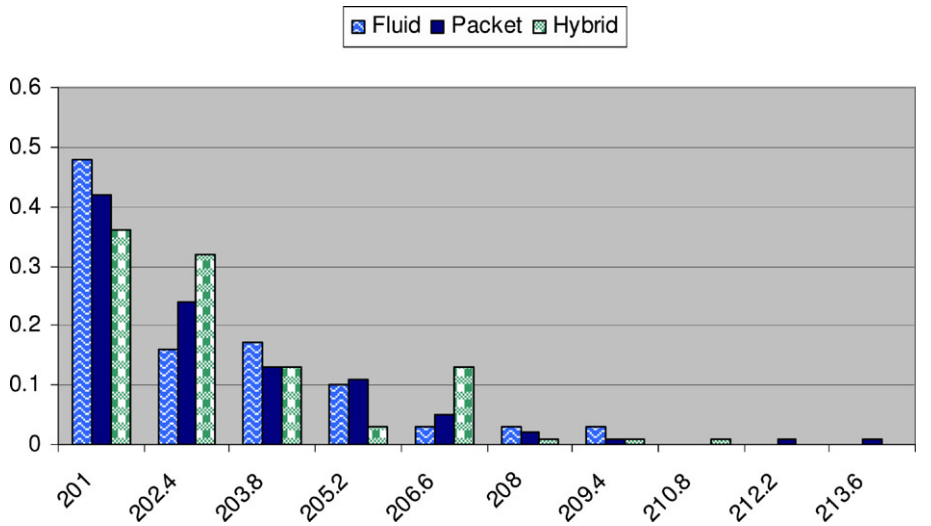

Fig. 14. Histograms of stopping times for IPA derivatives of loss rate as a function of an arrival rate parameter.

Table 11
Pure-packet version comparison of IPA derivatives subject to *HSR* and *MSR*

IPA derivative	T_H	HSR	MSR stopping time	MSR (% deviation)
Loss rate as a function of buffer size	600	−1.12054	209.8461	−1.04839 (−6.44%)
Workload time average as a function of buffer size	600	0.22901	201.9969	0.22250 (−2.84%)
Loss rate as a function of a service rate parameter	600	−0.21806	201.5469	−0.20813 (−4.56%)
Workload time average as a function of a service rate parameter	600	−0.00659	201.9969	−0.00676 (2.54%)
Loss rate as a function of an arrival rate parameter	600	0.21706	201.9969	0.20845 (−3.97%)
Workload time average as a function of an arrival rate parameter	600	0.00659	201.9303	0.00676 (2.48%)

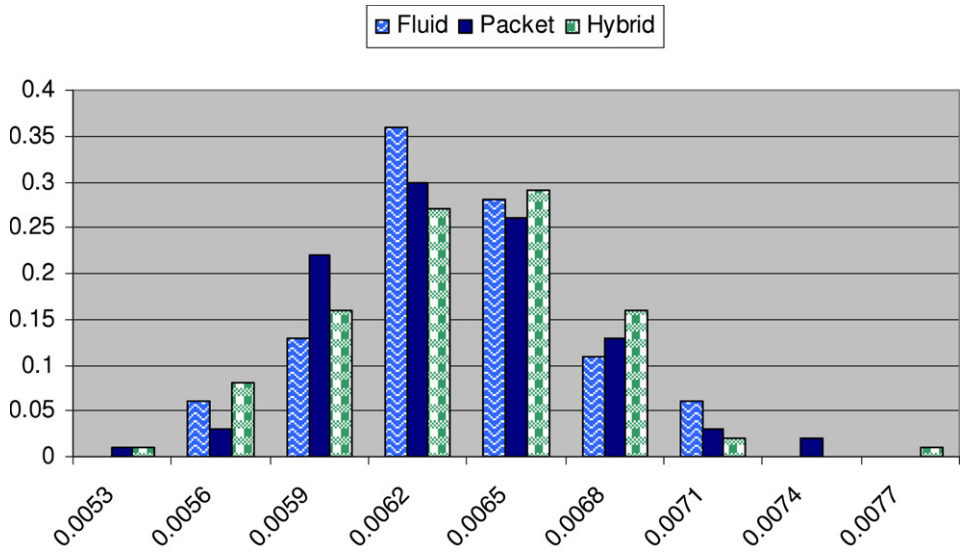

Fig. 15. Histograms of IPA derivatives of workload time average as a function of an arrival rate parameter.

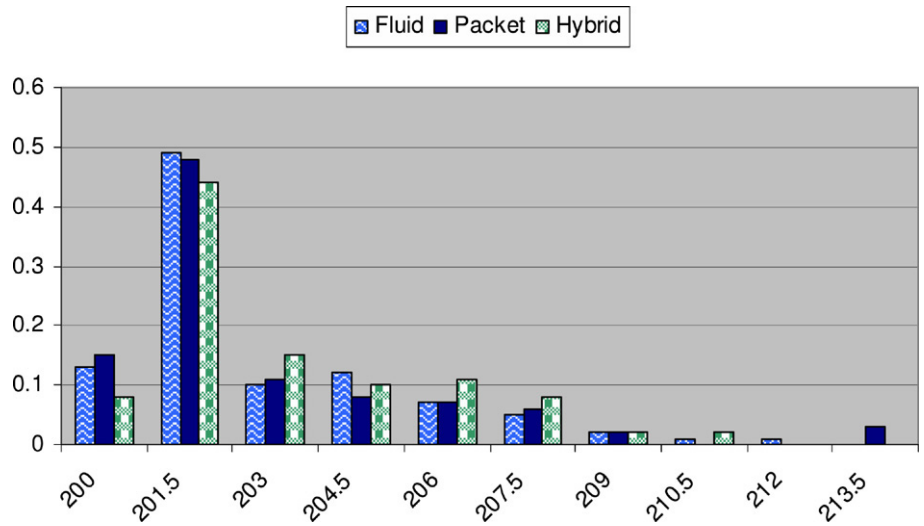

Fig. 16. Histograms of stopping times for IPA derivatives of workload time average as a function of an arrival rate parameter.

3. The *MSR* stopping rule is superior to the *HSR* stopping rule in that it provides comparable computational accuracy at a fraction of the computational complexity. This conclusion is supported by the study in Section 5.4, which compared compatible replications with different stopping rules.

The study reported in this paper can be extended in two directions. The first direction is to extend the empirical study of this paper to other queueing systems, especially those with feedback. For example, [23–25] derive IPA gradients in a general queueing setting, while [26] and [27] derive them for make-to-stock production-inventory systems with backorders and lost sales, respectively. We believe that this paper's results would extend to this type of system and other transaction-oriented systems.

The second and more difficult direction is to devise and experiment with optimization and control schemes that take advantage of both performance metrics and their IPA gradients. Work on these research directions will be reported elsewhere.

Table 12
Hybrid version comparison of IPA derivatives subject to *HSR* and *MSR*

IPA derivative	T_H	HSR	MSR stopping time	MSR (% deviation)
Loss rate as a function of buffer size	600	−1.11888	209.8461	−1.04839 (−6.30%)
Workload time average as a function of buffer size	600	0.22905	201.6441	0.22145 (−3.31%)
Loss rate as a function of a service rate parameter	600	−0.21843	201.2899	−0.20864 (−4.48%)
Workload time average as a function of a service rate parameter	600	−0.00657	201.6441	−0.00671 (2.15%)
Loss rate as a function of an arrival rate parameter	600	0.21828	201.6441	0.20834 (−4.55%)
Workload time average as a function of an arrival rate parameter	600	0.00657	201.8247	0.00672 (2.28%)

References

[1] P. Bratley, B.L. Fox, L.E. Schrage, A Guide to Simulation, Springer-Verlag, New York, NY, 1987.
[2] A.M. Law, W.D. Kelton, Simulation Modeling & Analysis, McGraw-Hill, 1991.
[3] D. Anick, D. Mitra, M.M. Sondhi, Stochastic theory of a data-handling system with multiple sources, The Bell System Technical Journal 61 (1982) 1871–1894.
[4] H. Kobayashi, Q. Ren, A mathematical theory for transient analysis of communications networks, IEICE Transactions on Communications E75-B (1992) 1266–1276.
[5] E.A. Hall, Internet Core Protocols: The Definitive Guide, O'Reilly & Associates, Inc., 2000.
[6] O. Kyas, ATM Networks, International Thomson Computer Press, 1996.
[7] G. Kesidis, A. Singh, D. Cheung, W.W. Kwok, Feasibility of fluid-driven simulation for ATM networks, in: Proceedings of IEEE GLOBECOM, vol. 3, 1996, pp. 2013–2017.
[8] N. Milidrag, G. Kesidis, M. Devetsikiotis, An overview of fluid-based quick simulation techniques for large packet-switched communication networks, in: Proc. SPIE ITCOM, Denver, CO, 2001.
[9] D.M. Nicol, Discrete event fluid modeling of TCP, in: Winter Simulation Conference, WSC 01, Arlington, Virginia, 2001.
[10] B. Liu, Y. Guo, J. Kurose, D. Towsley, W.B. Gong, Fluid simulation of large scale networks: Issues and tradeoffs, in: Proc. Intl. Conf. on Parallel and Distributed Processing Techniques and Applications, Las Vegas, Nevada, 1999.
[11] Y.C. Ho, X.R. Cao, Perturbation Analysis of Discrete Event Dynamic Systems, Kluwer Academic Publishers, Boston, MA, 1991.
[12] C.G. Cassandras, S. Lafortune, Introduction to Discrete Event Systems, Kluwer Academic Publishers, Boston, MA, 1999.
[13] L. Kleinrock, Queueing Systems, Vol. 1: Theory, Wiley, New York, NY, 1975.
[14] Y. Wardi, B. Melamed, Variational bounds and sensitivity analysis of continuous flow models, Journal of Discrete Event Dynamic Systems 11 (3) (2001) 249–282.
[15] C.G. Cassandras, G. Sun, C.G. Panayiotou, Stochastic fluid models for control and optimization of systems with quality of service requirements, in: Proc. 40th IEEE Conference on Decision and Control, CDC01, Orlando, Florida, 2001.
[16] Y. Wardi, B. Melamed, C.G. Cassandras, C.G. Panayiotou, On-line IPA gradient estimators in stochastic continuous fluid models, Journal of Optimization Theory and Applications 115 (2) (2002) 369–405.
[17] C.G. Cassandras, Y. Ward, B. Melamed, G. Sun, C.G. Panayiotou, Perturbation analysis for on-line control and optimization of stochastic fluid models, IEEE Transactions on Automatic Control 47 (8) (2002) 1234–1248.
[18] C.G. Cassandras, G. Sun, C.G. Panayiotou, Y. Wardi, Perturbation analysis and control of two-class stochastic fluid models for communications networks, IEEE Transactions on Automatic Control 48 (2003) 770–782.
[19] G. Sun, C.G. Cassandras, Y. Wardi, C.G. Panayiotou, G. Riley, Perturbation analysis and optimization of stochastic flow networks, IEEE Transactions on Automatic Control 49 (12) (2004) 2143–2159.
[20] B. Melamed, S. Pan, Y. Wardi, HNS: A streamlined hybrid network simulator, ACM Transactions on Modeling and Computer Simulation 14 (3) (2004) 3251–3277.
[21] S. Pan, Hybrid Network Simulation, Ph.D. Thesis, Rutgers Center for Operations Research (RUTCOR), Rutgers University, 2005.
[22] L. Breiman, Probability, Addison-Wesley, Reading, MA, 1968.
[23] Y. Wardi, G. Riley, IPA for spillover volume in a fluid queue with retransmissions, in: Proc. 43-rd IEEE Conference on Decision and Control, Nassau, Bahamas, 2004, pp. 3756–3761.
[24] H. Yu, C.G. Cassandras, Perturbation analysis of feedback-controlled stochastic flow systems, IEEE Transactions on Automatic Control 49 (8) (2004) 1317–1332.

[25] H. Yu, C.G. Cassandras, A new paradigm for an on-line management of communication networks with multiplicative feedback control, in: A. Girard, B. Sanso, F. Vasquez-Abad (Eds.), Performance Evaluation and Planning Methods for the Next Generation Internet, Springer-Verlag, 2005, pp. 297–332.
[26] Y. Zhao, B. Melamed, IPA derivatives for make-to-stock production-inventory systems with backorders, Methodology and Computing in Applied Probability 8 (2) (2006) 191–222.
[27] Y. Zhao, B. Melamed, IPA derivatives for make-to-stock production-inventory systems with lost sales, IEEE Transactions on Automatic Control (2007) (in press).